

Adaptive Mesh Refinement Computation of Solidification Microstructures using Dynamic Data Structures

Nikolas Provatas^{1,2}, Nigel Goldenfeld¹, and Jonathan Dantzig²

¹ *University of Illinois at Urbana-Champaign, Department of Physics*

1110 West Green Street, Urbana, IL, 61801

² *University of Illinois at Urbana-Champaign, Department of Mechanical and Industrial Engineering*

1206 West Green Street, Urbana, IL, 61801

(February 1, 2008)

We study the evolution of solidification microstructures using a phase-field model computed on an adaptive, finite element grid. We discuss the details of our algorithm and show that it greatly reduces the computational cost of solving the phase-field model at low undercooling. In particular we show that the computational complexity of solving any phase-boundary problem scales with the interface arclength when using an adapting mesh. Moreover, the use of dynamic data structures allows us to simulate system sizes corresponding to experimental conditions, which would otherwise require lattices greater than $2^{17} \times 2^{17}$ elements. We examine the convergence properties of our algorithm. We also present two dimensional, time-dependent calculations of dendritic evolution, with and without surface tension anisotropy. We benchmark our results for dendritic growth with microscopic solvability theory, finding them to be in good agreement with theory for high undercoolings. At low undercooling, however, we obtain higher values of velocity than solvability theory at low undercooling, where transients dominate, in accord with a heuristic criterion which we derive.

I. INTRODUCTION

Modeling solidification microstructures has become an area of intense study in recent years. The properties of large scale cast products, ranging from automobile engine blocks to aircraft components and other industrial applications, are strongly dependent on the physics that occur at the mesoscopic and microscopic length scales during solidification. The main ingredient of the solidification microstructure is the dendrite, a snowflake-like pattern of solid around which solidification proceeds. The microscopic properties of such cast products are determined by the length scales of these dendrites, and for this reason understanding the mechanisms for pattern selection in dendritic growth has attracted a great deal of interest from the experimental and theoretical community. In particular, a great deal of research has been undertaken to understand such issues as dendrite morphology, shape

and speed. Experiments on dendrite evolution by Glicksman and coworkers on succinonitrile (SCN) [1,2], and more recently Pivalic Acid (PVM) [3], as well as other transparent analogues of metals have provided tests of theories of dendritic growth, and have stimulated considerable theoretical progress [4–6]. These experiments have clearly demonstrated that in certain parameter ranges the physics of the dendrite tip can be characterized by a steady value for the dendrite tip velocity, radius of curvature and shape. Away from the tip the time-dependent dendrite exhibits the characteristic sidebranching as it propagates.

The earliest theories of dendritic growth solved for the diffusion field around a self-similar body of revolution propagating at constant speed [7,8]. In these studies the diffusion field is found to determine the product of the dendrite velocity and tip radius, but neither quantity by itself. Adding capillarity effects to the theory predicts a unique maximum growth speed [9]. Experiments, however, have shown that these theories do not represent the correct operating state for real dendrites.

The introduction of local models of solidification brought further insight to the steady state dendrite problem [10–13]. These models describe the evolution of the interface, incorporating the physics of the bulk phases into the governing equation of motion of the interface. A remarkable breakthrough of these models was to show that a steady-state dendrite velocity is obtained *only* if a source of anisotropy (e.g., in the interfacial energy) is present during dendritic evolution. The dendrite steady-state tip velocities appear in a discrete rather than continuous spectrum of values, making the role of anisotropy of great importance in the description of the dendrite problem, both in the local models and the full moving boundary problem [6,14,15]. It was further shown that only the fastest of a spectrum of steady state velocities is stable, thus forming the operating state of the dendrite. This body of theoretical work is generally known as *microscopic solvability theory*.

Dendritic sidebranching is widely believed to be caused by thermal fluctuations, which enter solidification models in the form of random noise possessing specific features [16–18]. However, the precise mechanism of sidebranch

amplification does not seem to be fully understood. It would appear that the manner in which thermal noise is amplified may depend on the overall dendrite morphology. For instance, it was shown that noisy fluctuations traveling down a paraboloid of revolution do not produce sidebranch amplitudes consistent with experiments [16], while fluctuations traveling down an initially missile-shaped dendrite amplify into sidebranches comparable to some experiments [18]. Karma also investigated the addition of interface fluctuations [17]. However, this source of noise only becomes relevant at high velocities.

The theoretical foundation around which most theories of solidification are based is the time-dependent Stefan problem. This theory describes the evolution of the thermal or solutal diffusion field around the solidification front, along with two accompanying boundary conditions. The first boundary condition relates the velocity of the moving front to the difference in thermal fluxes across the solid-liquid interface. The second, called the Gibbs-Thomson condition, relates the interfacial temperature to the thermodynamic equilibrium temperature, the local interfacial curvature and interface kinetics. The interface kinetics term adds a non-equilibrium correction to the interface temperature, usually assumed to be in local equilibrium for a given curvature. Solving the Stefan problem numerically has traditionally involved front tracking and lattice deformation to contain the interface at predefined locations on the grid [19,20]. This method is generally complicated to implement accurately and requires much effort. Moreover, it can be inefficient in handling coalescence of two or more interfaces.

The solution of the Stefan problem has been made more tractable with the introduction of the *phase-field* model. The phase-field model avoids this problem of front tracking by introducing an auxiliary continuous order parameter $\phi(\mathbf{r})$ that couples to the evolution of the thermal or solutal field. The phase field interpolates between the solid and liquid phases, attaining two different constant values in either phase, (e.g., ± 1) with a rapid transition region in the vicinity of the solidification front. The level set of $\phi(\mathbf{r}) = 0$ is identified with the solidification front, and the subsequent dynamics of ϕ are designed to follow the evolving solidification front in a manner that reproduces the Stefan problem [21–31].

The price to be paid for the convenience of the order parameter is the introduction of a new length scale W which represents a boundary layer over which the order parameter changes sign. This distance is referred to as the interface width, and does not appear in the Stefan problem. As such, one requirement of the phase-field model is to recover the Stefan limit in a manner that is independent of the interface width as W approaches some appropriate limit. considerable work has been done to relate W to various parameters of the phase-field model in order to establish a mapping between the phase-field model and the Stefan problem [22,32,33]. While the formal nature of these mappings does not seem to be very sensitive to the precise form of the phase-field model [33],

different asymptotic limits of the phase-field parameters can lead to widely varying complexity in the numerical implementation.

The introduction of the interface width W makes the phase-field model prohibitively expensive to simulate numerically for large systems, since the grid spacing must be small enough everywhere that the phase-field model converges to the sharp interface limit [22,32]. Caginalp and Chen [34] showed rigorously that the phase field model converges to the sharp interface limit when the interface width (and hence the grid spacing) is much smaller than the capillary length. While this result is necessary to establish that the phase-field model does map onto the Stefan problem, the parameter values required to realize the asymptotic limit can be computationally intractable. Experimentally, the physical sizes required to contain realistic microstructures can be many times the size of the thermal diffusion length, which in turn can be orders of magnitude greater than W . Thus, since $\Delta x_{\min} < W$, computing in the limit of a $W \rightarrow 0$ does not allow one to simulate experimental systems.

Recently Karma and Rappel [32] presented a different asymptotic analysis performed in powers of the ratio of the interface width to the diffusion length α/V_n , taken to be equal in both phases. Their procedure offers two computational advantages. The first is that it allows one to simulate the phase-field model with zero interface kinetics, *without* the need to make $W \rightarrow 0$. Specifically, this limit, as well as a non-zero kinetics limit, can be simulated with an interface width W (and hence the grid spacing) of order the capillary length, a much more tractable regime. Simulating solidification microstructures in the limit of zero interface kinetics is important because most experiments performed at low undercooling in materials such as succinonitrile are in this limit [2]. Karma and Rappel tested their asymptotics by comparing their simulations to the results of microscopic solvability theory, finding excellent agreement down to dimensionless undercoolings as low as 0.25.

A recent extension of Karma and Rappel’s analysis by Almgren [35] also promises to allow similar asymptotics to be performed on a two-sided model of solidification [35], i.e. when the diffusivities in the solid and liquid differ, relevant in the study of directional solidification of binary mixtures.

The theory of level sets [36,37] has also recently re-emerged as another effective tool that shows great potential in modeling dendritic growth. While related to the phase-field model, level-set theory does not require the presence of a thin interfacial with W , thus greatly reducing the stringent grid requirements posed by conventional phase-field models. To date, however, level set methods have not been benchmarked with solvability theory or other theoretical prediction for Stefan problems.

While expanding the horizon of solidification modeling, phase-field modeling has still been limited to small systems sizes, even when solved by adaptive algorithms [38]. The main problem is the presence of an interface

region with a minimal length scale that must be resolved. For a typical microstructures grown at dimensionless undercooling of 0.1 or less, the ratio of the system size to this minimal grid spacing can be greater than 2^{17} . With this restriction most numerical methods will naturally fail. What is needed to go beyond this limitation is an effective adaptive technique [38–41] which dynamically coarsens the grid spacing away from the front.

In this paper we present a new, computationally efficient adaptive-grid algorithm for solving a class of phase-field models suitable for the study of phase-boundary evolution. We study dendritic solidification modeled using two coupled fields, one for the order parameter and the other for the thermal field. Our algorithm effectively combines and implements ideas of adaptive-mesh refinement with ideas of dynamic data structures, allowing us to enlarge the window of large-scale solidification modeling.

The outline of this paper is as follows: In section one we introduce the physical model to be examined, summarizing its properties and its various limits. In section two we present a detailed description of our algorithm. In section three we present results on CPU-scalability of our algorithm and examine issues of grid convergence and grid anisotropy on our solutions. In section four we present results of dendritic growth with and without the presence of anisotropy in the surface energy. We show that for high undercooling, dendrites grown with our method converge to tip speeds in agreement with microscopic solvability theory. At low undercooling, however, we do not find agreement with steady state solvability theory, owing to long-lived transients in the thermal field evolution. In section four we conclude and discuss our results.

II. THE PHASE-FIELD MODEL

We model solidification using a standard form of phase-field equations which couple a thermal field to an order parameter field ϕ via a double well potential [32,22]. We begin by rescaling the temperature field T by $U = c_P(T - T_M)/L$, where c_P is the specific heat at constant pressure, L is the latent heat of fusion and T_M is the melting temperature. The order parameter is defined by ϕ , where we define $\phi = 1$ in the solid phase and $\phi = -1$ in the liquid phase. The interface is defined by $\phi = 0$. We rescale time throughout by τ_o , a time characterizing atomic movement in the interface. Length is rescaled by W_o , a length characterizing the liquid–solid interface. With these definitions, the model is written as

$$\begin{aligned} \frac{\partial U}{\partial t} &= D \nabla^2 U + \frac{1}{2} \frac{\partial \phi}{\partial t} \\ A^2(\vec{n}) \frac{\partial \phi}{\partial t} &= \nabla \cdot (A^2(\vec{n}) \nabla \phi) + g'(\phi) - \lambda U P'(\phi) \\ &+ \frac{\partial}{\partial x} \left(|\nabla \phi|^2 A(\vec{n}) \frac{\partial A(\vec{n})}{\partial \phi_x} \right) + \frac{\partial}{\partial y} \left(|\nabla \phi|^2 A(\vec{n}) \frac{\partial A(\vec{n})}{\partial \phi_y} \right), \end{aligned} \quad (1)$$

where $D = \alpha \tau_o / W_o^2$ and α is the thermal diffusivity. The function $f(\phi, U; \lambda) = g'(\phi) - \lambda U P'(\phi)$ is the derivative of the double-well potential with respect to ϕ and couples the U and ϕ fields via the constant λ . The primes on the functions $g(\phi)$ and $P(\phi)$ denote derivatives with respect to ϕ . Following Karma and Rappel [32], anisotropy has been introduced in Eqs. (1) by defining the width of the interface to be $W(\vec{n}) = W_o A(\vec{n})$ and the characteristic time by $\tau(\vec{n}) = \tau_o A^2(\vec{n})$, with $A(\vec{n}) \in [0, 1]$ given by

$$A(\vec{n}) = (1 - 3\epsilon) \left[1 + \frac{4\epsilon}{1 - 3\epsilon} \frac{(\phi_x)^4 + (\phi_y)^4}{|\nabla \phi|^4} \right]. \quad (2)$$

The vector

$$\vec{n} = \frac{\phi_x \hat{x} + \phi_y \hat{y}}{(\phi_x^2 + \phi_y^2)^{1/2}} \quad (3)$$

defines the normal to the contours of the ϕ field, where ϕ_x and ϕ_y are defined as the partial derivatives of ϕ with respect to x and y . The variable ϵ parameterizes the deviation of $W(\vec{n})$ from W_o and represents the anisotropy in the interface energy of the system. We note that this definition of anisotropy is not unique [33], but we expect results to be similar for other definitions of anisotropy.

In simulating the phase-field model we adopt the point of view that the order parameter field ϕ is a computational tool whose main purpose is to eliminate front tracking. As such we would like to simulate the model given by Eqs. (1) with W_o as large as possible. At the same time we would like the behavior of the model outside the boundary layer defined by ϕ to describe the Stefan problem as closely as possible. To this end, we relate the parameters of the phase-field model according to Ref. [32], valid in the asymptotic limit $W_o \ll \alpha/V_c$, where α/V_c is the diffusion length and V_c is a characteristic velocity of the front defined by ϕ .

The specific asymptotic limit we model is one where the U -field satisfies

$$\frac{\partial U}{\partial t} = D \nabla^2 U \quad (4)$$

everywhere away from the interface, while at the interface, the gradient of U satisfies

$$V_n = D \left(\left. \frac{\partial U}{\partial \vec{n}} \right|_{\vec{x}_{\text{int}}^-} - \left. \frac{\partial U}{\partial \vec{n}} \right|_{\vec{x}_{\text{int}}^+} \right), \quad (5)$$

where V_{int} is the velocity normal to the interface, denoted by \vec{x}_{int} . The notation \pm denotes the solid/liquid side of the interface, respectively. The description of the Stefan problem is completed by the Gibbs-Thomson condition and the interface kinetics condition

$$U(\vec{x}_{\text{int}}) = -d(\vec{n})\kappa - \beta(\vec{n})V_n, \quad (6)$$

where $d(\vec{n})$ is the capillary length, κ is the local curvature and $\beta(\vec{n})$ is the interface attachment kinetic coefficient,

all assumed to be in dimensionless form according to the above definitions. The capillary length is related to the parameters of Eqs. (1) by

$$d(\vec{n}) = a_1 \frac{W_o}{\lambda} [A(\vec{n}) + \partial_\theta^2 A(\vec{n})] \quad (7)$$

where $a_1 = 0.8839$ for the particular form of the free energy defined in Eqs. (1) [32] and θ is the angle between \vec{n} and the x -axis. The kinetic coefficient is given by

$$\beta = \frac{a_1 \tau_o}{\lambda W_o} \left[1 - \frac{\lambda a_2}{D} \right] \quad (8)$$

where $a_2 = 0.6267$ for our choice of the free energy functional citeKar95. One remarkable feature of Eqs. (7) and (8) is that W_o , τ_o and λ can be chosen to simulate arbitrary values of β , for W_o of order d_o . In particular, setting $\lambda = D/a_2$ allows us to compute the phase-field model in the limit of the Stefan problem [32], where $\beta = 0$. This is also an appropriate value for SCN, especially at low undercooling.

Equations (7) and (8) for β and d_o can be related to a wide class of free energies via the parameters a_1 and a_2 [32], which are related to integrals that depend on $g(\phi_o)$, $P(\phi_o)$ and $d\phi_o/dx$, where ϕ_o is the lowest order description of the order parameter field ϕ and is a solution of the equation

$$\frac{\partial^2 \phi_o}{\partial x^2} - \frac{dg(\phi_o)}{d\phi_o} = 0. \quad (9)$$

We also note that these asymptotics are a special case of a more general asymptotic analysis performed by Almgren [35], which relates the parameters of the phase-field model to those of the Stefan problem in the case of unequal diffusivities in the solid and liquid phases. In this case, the asymptotics provides an additional set of constraints on the admissible functions $P'(\phi)$, $g'(\phi)$, and hence a_1 and a_2 .

III. THE ADAPTIVE-GRID ALGORITHM

The main computational challenge of simulating Eqs. (1) involves resolving two competing length scales: the lattice spacing dx on which the simulation is performed and the physical size of the system L_B . Even with improved asymptotics, dx must remain relatively small, while L_B must be extremely large in order to make possible computations of extended solidification microstructures. Moreover, the main physics of solidification (and the evolution of most phase-boundary problems) occurs around an interface whose area is much smaller than the full computational domain. Near this interface the order parameter varies significantly, while away from the interface variations in ϕ are small. Meanwhile, the thermal field U extends well beyond the interface and has much more gradual variation in its gradients, permitting

a much coarser grid to be used to resolve U . The most obvious manner to overcome this problem is to use a method that places a high density of grid points where the interface of ϕ or U varies most rapidly and a much lower grid density in other regions. Furthermore, the method must dynamically adapt the grid to follow the evolving interface [38–41], while at the same time maintaining a certain level of solution quality.

We solve Eqs. (1) using the Galerkin finite element method on dynamically adapting grids of linear, isoparametric quadrilateral and triangular elements. The grid is adapted dynamically based on an error estimator that utilizes information from both the ϕ and U fields. We wrote our code in FORTRAN 90 (F90), taking advantage of the efficiency of FORTRAN 77 while using advanced C-like features, such as data structures, derived data types, pointers, dynamic memory allocation and modular design to conveniently adapt the grid and the solution fields.

In the broadest sense, our algorithm performs functions that can be divided into two classes. The first deals with the establishment, maintenance and updating of the finite element grids, and the second with evolving ϕ and U on these grid, according to Eqs. (1). We presently describe these classes, the adaptive grids, the finite element procedure, and the interconnections of these processes.

A. The Finite Element Grids

The first class of functions in our algorithm centers around maintaining a grid of finite elements on a data structure known as a *quadtree* [42–44] which replaces the standard concept of a uniform grid as a way of representing the simulational grid. The quadtree is a tree-like structure with branches up to a prespecified level. Branches of the quadtree are themselves data structures that contain information analogous to their parent, from which they branched, but one level down. Fig. 1 illustrates the structure of a quadtree as well as the relation between elements at different levels of refinement. Every entry on the quadtree contains information pertaining to a 4-noded isoparametric quadrilateral finite element. This information includes the following:

- values of ϕ and U at the four nodes
- the nodal coordinates of the element
- the level of refinement of the element on the quadtree
- the value of the current error estimate
- The element number, which contains information about the coordinates of the element and its level of refinement
- an array mapping the element's four nodes onto the

entries of a global solution array

- pointers to the element’s nearest neighbors sharing a common edge and at the same level of grid refinement
- a variable that determines whether or not an element contains further sub-elements which we term *child* elements
- pointers to an element’s child elements
- a pointer to the *parent* element from which an element originates

The elements of the quadtree can be refined by splitting into four child elements, each sharing the same parent element one level down on the quadtree and each with its own unique information, as outlined above. A parent element and its four child elements are referred to as a *family*. Refinement produces a finer mesh within the confines of the original parent grid by bisecting each side, as shown in Fig. (1). Unrefinement, which consists of fusing the four child elements back into the parent, has the opposite effect, locally creating a coarser mesh. Both refinement and unrefinement proceed via dynamic memory allocation, making our code scalable. We note that unrefinement can occur only if the child elements do not possess further child of their own. Also, in order to avoid having regions of very different refinement bordering each other, we impose the restriction that any two neighboring quadrilateral elements may be separated by no more than one level of refinement (see Fig. (1)). We define the level of refinement of an element by l_e such that a uniform grid at a refinement level l_e would contain $2^{l_e} \times 2^{l_e}$ grid points in a physical domain $L_B \times L_B$.

Special cases where an element has no children, a missing neighbor, or no parent are handled by null pointers. The latter case occurs only for the root of the quadtree.

All elements at a given level of refinement on the quadtree are “strung” together by a linked-list of pointers, referred to as the *G-lists*. There are as many *G-lists* as there are levels of refinement in the quadtree. Each pointer in the *G-list* points to (accesses) the location in memory assigned to one element of the quadtree. The purpose of the *G-list* is to allow traversal of the quadrilateral elements sequentially by level, rather than by recursively traversing quadtree from the root down, a procedure which is memory intensive and relatively slow.

Alongside the main grid of quadtree elements, the code maintains two independent grids representing special linear isoparametric triangular and rectangular elements. These elements are used to connect the extra nodes that arise when two or more quadrilateral elements of differing refinement levels border each other. These element types are referred to as *bridging elements*. They are maintained as two linked-lists of derived data types, one containing information about triangular elements and the other rectangular. Elements of both these grids include the follow-

ing information:

- the values of ϕ and T at the three nodes (four for rectangles) of the element
- the nodal coordinates
- node numbers that map the element’s nodes onto the global solution array

The types of bridging triangles and rectangles that can occur are enumerable and shown in Fig. (2).

The main set of operations performed on the grids described above concern refinement of the finite element mesh as a whole. The refinement process is performed only on the quadrilateral mesh. The triangular and rectangular grids are established after this process is completed (see Fig. (1)). To refine the grid the code traverses the elements of the quadtree, refining (unrefining) any element whose error function, discussed below, is above (below) a critical value $\sigma_h(\sigma_l)$. We also note that fusion of four quadrilateral elements can occur only if all four of its children’s error functions are below the critical value σ_l , where $\sigma_l < \sigma_h$. We found that if $\sigma_l = \sigma_h$ the grid sets into oscillations, where large numbers of elements become alternatively refined at one time step, then unrefined at the next.

The processes described thus far are grouped into modules that encapsulate various related tasks, and which can cross-reference each other’s data and instructions. The module highest up in the hierarchy contains the definition of the quadtree data structure and routines that construct the initial uniform grid, refine and unrefine individual quadrilateral elements, and set the initial conditions. Another module constructs the *G-lists*. It contains routines that construct the initial *G-list* from initial uniform quadtree data structure, as well as add or delete element pointers from the *G-list* as elements are created or deleted from the quadtree. Another module accessing both the previous ones’ data structures has the role of creating the triangular and rectangular element grids. It contains definitions for creating triangular and rectangular elements data structures and routines that search the quadtree, building the linked lists of triangles and rectangles that make up these grids. The main program is contained in its own module and contains the driver program that creates the initial grids, *G-lists* and triangular and rectangular element types. The driver program also sets into motion the final link in the simulation, which evolves ϕ and U and periodically adapts the dynamic grid by calling procedures described above. A flowchart of these processes is shown in Fig. (3).

B. The Finite Element Formulation

The integration of Eqs. (1) is done by the final module in the code. This module performs four main processes:

1. Maps the internal element node numbers to the indices of a global solution vector. The ϕ -field is mapped onto the odd numbers, $(1, 3, 5, \dots)$, while U is stored on the even numbers of the global solution vector $(2, 4, 6, \dots)$
2. Advances the U and ϕ field-vectors by N_r time steps on the finite element grids defined above
3. Calculates an error function for each element of the quadtree, based on error estimate of the quadrilateral elements
4. Invokes routines in the modules described above to refine the grid according to this error estimator

Steps (1)-(4) are repeated until a sufficient time evolution of the microstructure is established. The variable N_r is set such that the interface remains within the regions of fine mesh between regriddings, which we typically choose to be 100 time steps. Step (1) involves searching all elements, and their neighbors, and assigning each node a unique number, that will have a counterpart on a global solution vector.

The finite element discretization of Eqs. (1) is done using Galerkin's weighted residual method [45]. The method begins by assuming that ϕ and U are interpolated within an element as

$$\phi^e = \sum_{i=1}^N \phi_i^e N_i(x, y) \quad U^e = \sum_{i=1}^N U_i^e N_i(x, y) \quad (10)$$

where ϕ_i^e and U_i^e are the field values at the N nodes of the element e , and their interpolated values in its interior. The functions $N_i(x, y)$ are standard linear interpolation functions appropriate to the element being used [46], and satisfy

$$N_i(x_j, y_j) = \delta_{i,j}, \quad (11)$$

where $\delta_{i,j}$ is the Kronecker delta. Rewriting the differential equations for ϕ in Eqs. (1) as $L_\phi \phi = 0$, and of the U -equation as $L_U U = 0$, the Galerkin method requires that

$$\begin{aligned} \int_{\Omega_e} N_i(x, y) L_\phi \phi^e(x, y) dx dy &= 0 \\ \int_{\Omega_e} N_i(x, y) L_U U^e(x, y) dx dy &= 0, \end{aligned} \quad (12)$$

for $i = 1, 2, 3, \dots, N$, where Ω_e represents the area of an element e . Substituting Eqs. (10) into Eqs. (12), we obtain two linear algebraic equations for ϕ_i and U_i , $i = 1, 2, 3, \dots, N$ in the element e .

We next define $\{\Phi\}^e = (\phi_1, \phi_2, \phi_3, \dots, \phi_N)^T$ and $\{U\}^e = (U_1, U_2, U_3, \dots, U_N)^T$, where the superscript T denotes transpose, making $\{\Phi\}^e$ and $\{U\}^e$ column vectors. The linear algebraic statement of the finite element form of Eqs. (1) then becomes

$$[\hat{C}](\{\phi\}^e) \frac{d\{\phi\}^e}{dt} = ([M] + [E]) \{\phi\}_n^e + \{F; \lambda\}^e \quad (13)$$

$$[C] \frac{d\{U\}^e}{dt} = D[A]\{U\}^e + \frac{1}{2}[C] \frac{d\{\phi\}^e}{dt},$$

where the matrices $[C]$, $[\hat{C}]$, $[A]$, $[M]$ and $[E]$ and the vector $\{F; \lambda\}^e$ are given by

$$[C] = \int_{\Omega_e} [N]^T [N] dx dy, \quad (14)$$

$$[\hat{C}] = \int_{\Omega_e} [N]^T [N] A^2(\theta(\phi^e)) dx dy, \quad (15)$$

$$[A] = - \int_{\Omega_e} ([N]^T [N_x] + [N]^T [N_y]) dx dy, \quad (16)$$

$$[M] = - \int_{\Omega_e} ([N]^T [N_x] + [N]^T [N_y]) A^2(\theta(\phi^e)) dx dy, \quad (17)$$

$$[E] = - \int_{\Omega_e} ([N]^T [N_x] - [N]^T [N_y]) A(\theta(\phi^e)) \omega(\theta(\phi^e)) dx dy, \quad (18)$$

$$\{F; \lambda\}^e = \int_{\Omega_e} [N]^T f(\phi^e, U^e; \lambda) dx dy, \quad (19)$$

where $[N_x]$, $[N_y]$ denote the partial derivatives of the vector of shape functions with respect to x and y , respectively. The function A is just Eq. (2) rewritten in terms of the angle θ that the normal to the interface makes with the x -axis. Specifically, defining

$$\tan \theta(\phi^e) = \frac{\partial \phi_{i,y}^e}{\partial \phi_{i,x}^e}. \quad (20)$$

then

$$A(\theta(\phi^e)) = (1 - 3\epsilon) \left[1 + \frac{4\epsilon}{1 - 3\epsilon} \frac{(1 + \tan^4 \theta)}{(1 + \tan^2 \theta)^2} \right] \quad (21)$$

while $\omega(\theta)$ is proportional to the derivative of $A(\theta)$, and is given by

$$\omega(\theta(\phi^e)) = 16\epsilon \frac{\tan\theta(1 - \tan^2\theta)}{(1 + \tan^2\theta)^2}, \quad (22)$$

We use a lumped formulation for the matrices $[C]$ and $[\hat{C}]$ [45]. In this procedure, the row vector of shape functions, $[N]$ in Eq. (14) is replaced by the identity row vector $[I] = [1, 1, 1, \dots]$. The resulting matrix $[C]$ then consists of identical columns, each of which contains the element $N_i(x, y)$ in the position of the i^{th} row. A lumped term is then defined as a diagonal matrix whose entries take on the value

$$L_c = \frac{1}{4} \sum_{i=1}^{\text{nodes}} \int_{\Omega_e} N_i(x, y) dx dy. \quad (23)$$

The use of a lumped matrix for $[C]$ allows us to assemble a diagonal matrix for the left hand side Eqs. (13), stored as a one-dimensional vector rather than two-dimensional matrices that would be required if we used the consistent formulation for the assembly of the $[C]$ matrices. Indeed, microstructures evolving at low undercooling can produce interfaces with over 2×10^5 elements, making the storing of 2×10^{10} matrices impossible.

The global $\{\phi\}$ (obtained after assembly of the element equations in field in Eqs. (13)) is time-stepped using using a forward difference (explicit) time scheme. For each time step of the ϕ field, the global U field is then solved iteratively using a Crank-Nicholson scheme. Convergence of $\{U\}_{n+1}$ is obtained in a few iterations.

C. The Error Estimator

Regridding is based on an error estimator function, which is obtained following Zienkiewicz and Zhu [46], based on the differences between calculated and smoothed gradients of the ϕ and U fields. Specifically, we define the *composite field*

$$\Psi = \phi + \gamma U \quad (24)$$

where γ is a constant. We discuss the selection of γ in more detail below. This definition allows us to regrid according to the requirements of both the ϕ and U field, as opposed to using only gradients of the ϕ -field in establishing the grid [38]. Since it is ϕ and U that are being calculated, and not their gradients, we do not expect the gradient of Ψ to be continuous across element boundaries, due to the order of the interpolation used. Thus we expect the difference between the calculated and smoothed (continuous across element boundaries) gradients to provide a reasonable estimate of error. This method appropriately meshes regions of both steep gradients and regions where the ϕ and U fields change rapidly.

We define the error estimator function \vec{e} as

$$\vec{e} = \vec{q}_s - \vec{q}_c \quad (25)$$

where \vec{q}_c and \vec{q}_s are the calculated and smoothed gradients of Ψ respectively. The smoothed gradients are calculated to be continuous across element boundaries. To determine \vec{q}_s we assume it to be interpolated in the same way as the ϕ and U fields, namely

$$\vec{q}_s = [N]\{Q^s\} \quad (26)$$

where $[N]$ is the row vector of element shape functions, and $\{Q^s\}$ is a 4×2 matrix whose columns represent the nodal values of fluxes of Ψ in the x and y direction, respectively. To find $\{Q^s\}$ we use Galerkin's method, minimizing the weighted residual

$$\int_{\Omega_e} [N]^T \vec{e} d\Omega_e = \int_{\Omega_e} [N]^T ([N]\{Q^s\} - \vec{q}_c) d\Omega = 0 \quad (27)$$

The calculation is simplified by lumping the left hand side of Eq. (27), leading to

$$\left(\int_{\Omega_e} [N]^T [\mathbf{1}] d\Omega \right) \{Q^s\} = \int_{\Omega_e} [N]^T \vec{q}_c d\Omega, \quad (28)$$

where $[\mathbf{1}] = [1, 1, 1, \dots, N]$. Assembling Eq. (28) for all quadrilateral elements yields an equation for the smoothed gradients $\{Q\}^g$ of the global field Ψ , at all element nodes, of the form

$$[D]\{Q\}^g = b, \quad (29)$$

where $[D]$ is a diagonal matrix, due to “mass” lumping, and $\{Q\}^g$ is a $N \times 2$ matrix for the global, smoothed flux.

For the actual error updating on the elements of the quadtree we used the normalized error defined by

$$E_e^2 = \frac{\int_{\Omega_e} |(\vec{q}_s - \vec{q}_c)|^2}{\sum_e \int_{\Omega} |\vec{q}_s|^2}. \quad (30)$$

The domain of integration Ω in the denominator denotes the entire domain of the problem. Thus E_e^2 gives the contribution of the local element error relative to the total error calculated over the entire grid.

Fig. (4) shows a snapshot at 10^5 time steps into the a simulation of a thermal dendrite computed with our algorithm. The figure shows ϕ and U as well as the current grid. The dendrite is four-fold symmetric, grown in a quarter-infinite space, initiated by a small quarter disk of radius R_o centered at the origin. The order parameter is defined on an initially uniform grid to be its equilibrium value $\phi_o(\vec{x}) = -\tanh((|\vec{x}| - R_o)/\sqrt{2})$ along the interface. The initial temperature decays exponentially from $U = 0$ at the interface to $-\Delta$ as $\vec{x} \rightarrow \infty$. The parameters set for this simulation are $\Delta = 0.70$, $D = 2$, $dt = 0.016$ and λ chosen to simulate $\beta = 0$. The system size is 800×800 , with $\Delta x_{\min} = 0.4$, and about half of the computational domain in each direction is shown. Sidebranching is evident, and arises due to numerical noise. This simulation was completed in approximately 15 cpu-hours on a Sun UltraSPARC 2200 workstation.

IV. SCALABILITY AND CONVERGENCE PROPERTIES OF THE ADAPTIVE-GRID ALGORITHM

In this section we present results that illustrate the convergence properties of solutions of Eqs. (1) computed with our algorithm, the effect of grid-induced anisotropy of the adapting mesh, and the speed increase obtained by using an adapting grid.

A. CPU-Performance

We examined the cpu-scalability of our algorithm as a function of system size by growing dendrites in systems of various linear dimension L_B and measuring the cpu time required for the dendrite branches to traverse the entire system. Fig. 5 shows a plot of these data for a dendrite grown at undercooling $\Delta = 0.55$ using the same parameters as in Fig. 4. The minimum grid spacing has been set to $\Delta x_{\min} = 0.4$ in this data. Fig. 5, clearly shows that $R_t^a \sim L_B^2$. This relationship can be obtained analytically as follows.

The number of calculations performed, per simulation time step, is proportional to the number of elements in the grid. This relationship is set in turn by the arclength of the interface being simulated multiplied by the diffusion length D/V_n . This product defines the arclength over which the highest level of refinement occurs. For a needle-like dendrite, the arclength is approximately L_B . Moreover, since the dendrite tip moves at a constant velocity V_n , then

$$R_t^a = \left[\frac{R_o^a D}{V_n^2 \Delta x_m^2} \right] L_B^2, \quad (31)$$

where R_o^a is a constant that depends on the details of the implementation of the algorithm used to evolve Eqs. (1). The cpu time needed to compute the traversal time on a uniform grid, R_t^u , is found, by the same analysis, to be

$$R_t^u = \left[\frac{R_o^u}{V_n \Delta x_m^2} \right] L_B^3. \quad (32)$$

where R_o^u also depends on the implementation but is likely to be smaller than R_o^a . Thus, comparing our method with simulation on a uniform grid we obtain

$$\lim_{L_B \rightarrow \infty} R_t^a / R_t^u = \frac{1}{L_B}. \quad (33)$$

For larger systems, the adaptive scheme should always provide faster CPU performance regardless of implementation. Indeed, any method that uses a uniform grid of any sort, will eventually be limited by memory requirements as L_B becomes large. The arguments leading to Eq. (31) can also be generalized to any problem of evolving phase boundaries, always yielding the conclusion that

cpu time scales with arclength in the problem being considered. We note that when interface convolutions become of order $\Lambda \sim \Delta x_{\min}$, fine-grid regions separated by less than Λ will merge and the number of elements will stop growing locally. This makes the simulation of fractal-like patterns feasible as the arclength of the interface is bounded from above by $L_B \times L_B$. Finally, we note that adaptive gridding would especially improve the cpu performance of problems similar to spinodal decomposition, where the total interface decreases with time.

B. Induced Lattice Anisotropy

We tested the effective anisotropy of our dynamically adapting lattice in two independent ways. The first follows the method outlined by Karma [32]. We fix the temperature far from the interface to be constant T_∞ everywhere, initially setting it to a critical value at which the isotropic surface energy just balances the bulk free energy. For a specified background temperature, the crystal will only grow if its radius is greater than a critical value R_o . The radius R_o can be related to the background temperature through the total Gibbs free energy of the system, given by

$$\Delta G = -\pi r^2 \frac{L \Delta T}{T_M} + 2\pi r \sigma, \quad (34)$$

where L is the latent heat of fusion, $\Delta T = T_m - T_\infty$, where T_m is the melting temperature, T_∞ is the temperature far away from the interface, and σ is the surface tension. Minimizing ΔG with respect to r yields R_o as a function of δT as

$$R^* = d_o / \Delta T, \quad (35)$$

where d_o is the capillary length defined as $d_o = 2\sigma T_M / L$.

One finds an equilibrium shape of the interface when the background temperature field ΔT (written in terms of U) is adjusted dynamically so as to maintain the velocity of the interface at zero as measured long the x -axis. Thus, ΔT is increased if the velocity decreases, and decreased if it grows. The effective anisotropy is inferred by fitting the computed interface to an equation of the form

$$R(\theta) = R_o(1 + \epsilon_{\text{eff}} \cos \theta), \quad (36)$$

where $R(\theta)$ is the radial distance from the center of the crystal to its interface and θ the polar angle. The effective anisotropy ϵ_{eff} represents the modification of the anisotropy ϵ due to the grid. Fig. (6) illustrates a crystal grown to equilibrium using an input anisotropy $\epsilon = 0.04$. Using Eq. (36) we found $\epsilon_{\text{eff}} = 0.041$, within 5% of ϵ . Similar accuracy was found for $\epsilon = 0.02, 0.03$ and 0.05 .

We also tested for grid anisotropy by rotating the grid by 45 degrees, which should represent the lowest accuracy for square elements. We compared the tip speed

of dendrites grown in this direction to that of dendrites whose principal growth direction is along the x -axis. Fig. 7 shows the tip velocity for the case of a dendrite grown at $\Delta = 0.55$ ($\epsilon = 0.05$, $D = 2$, $\beta = 0$, $dt = 0.016$, $\Delta x_{\min} = 0.4$) compared with the same case when growth occurs along the x -axis. The tip velocity approaches an asymptotic value that is within approximately 5% of the tip velocity computed when the anisotropy is aligned with the x -direction.

C. Convergence and Grid Resolution

We tested the convergence of solutions as a function of the minimum grid spacing Δx_{\min} . We used an undercooling of $\Delta = 0.55$, with $D = 2$, $dt = 0.016$, $\Delta x_{\min} = 0.4$, and set λ to simulate $\beta = 0$. The parameter $\gamma = 1.8$, which assured that regions of rapid change of ϕ and U were always encompassed in the regions of highest grid resolution. We examined the tip speed of a dendrite for $0.3 \leq \Delta x_{\min} \leq 1.6$, finding relatively good convergence of the tip speed to theoretical prediction of microscopic solvability theory discussed above. Fig. (8) shows the asymptotic steady state tip velocity for each case, superimposed on the solid line, which is the result of solvability theory for $\Delta = 0.55$. It is surprising that the solution convergence is rather good even for $\Delta x_{\min} = 1.6$. We have found similar convergence properties for the case of $\Delta = 0.25$. Specifically, using $\Delta x_{\min} = 0.4$ and $\Delta x_{\min} = 0.78$ gives essentially identical results.

The introduction γ in the error function Ψ gives us the freedom to tune the degree to which the fine grid layering encompasses the thermal field as well as the ϕ field. Setting $\gamma = 0$ leads to a uniform-like mesh at the highest level of refinement *only* around the most rapidly changing regions of ϕ , while the U field becomes encompassed in a rather disorderly combination of quadrilateral and triangular elements. We found that this effect can increase the tip-speed error by several percent, as well as increase fluctuations in tip speed. Increasing γ produces a smooth layering of coarser uniform-like meshes ahead of the ϕ -field, corresponding to region of large gradients in U . Fig. 9 compares the mesh around the tip of a dendrite grown at $\Delta = 0.65$ for $\gamma = 0$ and $\gamma = 4$. The figure illustrates the gradual mesh layering encompassing the thermal field for $\gamma = 4$. In Fig. (9), $D = 1$, $dt = 0.016$, $\Delta x_{\min} = 0.4$ and λ is chosen to simulate $\beta = 0$. Fig. 7 also shows the tip speed for $\Delta = 0.55$ for the cases $\gamma = 0$ and 1.8, while Fig. (10) shows the tip velocity for a dendrite grown at $\Delta = 0.3$ with $\gamma = 0$ and 20, respectively. In Fig. (10) $D = 10$, $\Delta x_{\min} = 0.4$, $dt = 0.048$ and $\beta = 0$. In this case the higher value for γ allows the tip velocity to approach within approximately 5% of the solvability answer, as in Ref. [32]. Raising γ further does not produce any further changes in tip speed.

V. DENDRITIC GROWTH USING ADAPTIVE GRIDDING

In this section we present results for two dimensional solidification with and without interface anisotropy. We illustrate the robustness of our algorithm and use it, in particular, to investigate dendritic growth at low undercooling, presenting new results on dendrite tip-speed selection.

A. Dendritic Growth without Surface Tension Anisotropy

When the anisotropy parameter ϵ in Eqs. (1) is set to zero, solidification proceeds without the emergence of any preferred direction. In this case it is well known that a seed crystal larger than a critical radius will eventually grow to become unstable to fluctuations, and will break into surface undulations via the Mullins-Sekerka instability [20]. Fig. 11 shows a series of time steps in the evolution of a solidifying disk grown at $\epsilon = 0$, $\Delta = 0.65$, $D = 4$, and λ set to generate $\beta = 0$, making $d_o = 0.1385$. We use 11 levels of refinement and an 800×800 system, making $\Delta x_{\min} = 0.4$. For coarser meshes the Mullins-Sekerka instability sets in sooner due to grid noise. As Δx_{\min} is made smaller, grid noise becomes smaller, and one must wait longer for the true “thermal noise” to set in and make the crystal interface unstable. The dynamically evolving grids are also shown. This figure clearly demonstrates how the grid creation scales with the arc-length of the solidifying surface.

B. Dendritic Growth with Surface Tension Anisotropy

When surface tension anisotropy is present a crystalizing disk forms dendritic branches which travel along the symmetry axes of the anisotropy, driven by the anisotropy to a steady state tip velocity [6,11–15]. As a verification of our algorithm we measured tip-velocities and shapes for dendrites grown at intermediate undercoolings. Fig. 12 shows the dimensionless tip velocity (Vd_o/D) versus time for $\Delta = 0.45$ and 0.65 and $\epsilon = 0.05$. In Fig. (12) the dimensionless diffusivities $D = 3$ and 1 and the dimensionless capillary length are $d_o = 0.186$ and 0.544, respectively. In both cases λ has been set to simulate $\beta = 0$ kinetics at the interface, while $\gamma = 4$ and 1.8, respectively. These values of γ are chosen so as to minimize grid-layering error. The solid horizontal lines represent the theoretical values obtained from microscopic solvability theory. In all cases the converged velocities are within a few percent of the theoretical prediction. The case of $\Delta = 0.65$ includes data for three systems sizes. These results of system size are typical for intermediate Δ , showing a relatively rapid leveling to

an asymptotic speed to within a few percent of that predicted by solvability theory. Fig. (13) also shows a plot of the dendrite tip shapes produced by our simulations, superimposed on the shapes predicted by solvability theory.

C. Dendritic Growth at Low Undercooling

At lower undercooling we encounter significant finite-size effects which cause the tip velocity to deviate from the solvability prediction. Fig. 14 shows the evolution of the tip-velocity for $\Delta = 0.25$ in two different system sizes. For a system of size $L_x = 6400 \times L_y = 400$, the velocity goes to within a few percent of the solvability prediction. For a system size 6400×3200 the tip velocity seems to settle close to a value that exceeds the solvability prediction by 8%. This effect is even larger at $\Delta = 0.1$, also shown in Fig. (14), where the tip speed approaches a value about 3 times larger than that predicted by solvability theory.

To understand this finite-size dependence of tip velocity at low undercooling, we note that at low Δ , the thermal fields of the two dendrite branches overlap, producing a thermal envelope very different from that which emerges for the single, isolated dendrite branch assumed in solvability theory. At large undercooling, each dendrite arm quickly outruns the other's thermal boundary layer, and solvability theory should apply, as is seen in Fig. (4) where $\Delta = 0.7$. The conditions of solvability theory can be better approximated at lower undercooling if simulations are performed in a domain which is small in one direction. For the simulation performed with $\Delta = 0.25$ in a small box (6400×400), the branch in the y-direction is extinguished by its interaction with the wall and the velocity quickly approaches the solvability prediction. However, when both branches are present, as in the simulation with $\Delta = 0.25$ in the larger box (6400×3200), their interaction leads to an increased tip-velocity because the dendrites are embedded in a circular rather than parabolic diffusion field.

This is also clearly seen for a dendrite growing at $\Delta = 0.1$ in Fig. (15), where the dendrite shape and its associated field are shown for $\Delta = 0.1$ ($D = 13$, $d_0 = 0.043$, $\epsilon = 0.05$, $\Delta x = 0.78$, $dt = 0.08$). The dendrite arms never became free of each other in this simulation, causing the observed deviation from solvability theory shown in Fig. (14). We note that to avoid having the thermal field feel the effect of the sides of the box we perform our simulations in computational domains for which $L_x \sim (5 - 10)D/V_n$. To meet this criterion the simulation for $\Delta = 0.1$ was performed in a 102400×51200 domain, which is about $10D/V_n$. We note that the ratio of the largest to smallest element size in this simulation is 2^{17} . A fixed mesh having the same resolution everywhere would contain 9×10^9 grid points.

We can estimate the time t^* when the growth of the

dendrite tip crosses over from the transient regime where the branches interact to that where they become independent by equating the length of the full diffusion field, $3(Dt^*)^{1/2}$, to the length of a dendrite arm, $V_n t^*$. This gives the crossover time as

$$t^* = 9D/V_n^2. \quad (37)$$

The values for t^* corresponding to the cases $\Delta = 0.45, 0.55, 0.65$, and $\Delta = 0.25$ and 0.10 in Figs. (7, 12 and 14 confirm this scaling.

These results at low undercooling have important implications when comparing theory to experimental observations. In particular, since the transient time $t^* \rightarrow \infty$ as $\Delta \rightarrow 0$, it does not appear likely that independent predictions for tip speed and radius, as given by solvability, are likely to be observed experimentally. In this regime, the appropriate theory to use to obtain predictions of the tip speed and velocity is one which explicitly takes into account the long range effects of interacting thermal fields of other branches. Almgren, et al. present one such approach [47]. In particular, study of real dendrites with sidebranches, growing at low undercooling, will require such treatment. In closing, we note that while the independent predictions of tip speed and radius deviate from that of solvability theory at low undercoolings, the dimensionless *stability parameter* $\sigma^* = 2d_0 D/V_n R^2$ does agree within a few percent to solvability theory.

Further investigation of the tip speeds at low undercooling, comparison with experiments and new results for two-sided directional solidification will be reported in forthcoming publications.

VI. CONCLUSION

In this paper we present an efficient algorithm used to study solidification microstructures by adaptive refinement on a finite element mesh, and solving the phase-field model given by Eqs. (1). Our algorithm was made particularly robust by using dynamic data structures and pointer variables to represent our evolving grid. As well, the modular nature of our code offers an efficient method of expanding the code to different situations.

We found that our solution time scales with the arc-length of the interface being simulated, allowing simulation of much larger systems and at very low undercoolings. In particular, simulations for undercoolings as low as $\Delta = 0.1$ are quite straightforward in systems larger than $10D/V_n$. This undercooling represents the upper limit of dendrite growth in experiments [2].

Dendrite tip-velocities at intermediate to high undercoolings were found to agree with solvability theory to within a few percent. At low Δ , we found that the transient interaction of thermal fields from perpendicular dendrite branches modifies the tip-velocity from that given by solvability theory at times shorter than an estimated crossover time. Since this crossover time itself

becomes larger as Δ decreases, it is likely that transient effects will play a leading role in determining the tip velocity at low undercooling. Furthermore, this suggests that at low Δ the tip-velocity in the presence of side-branching will be different than that predicted by solvability theory.

Our algorithm is currently being used to examine directional solidification in models with unequal diffusivities in the solid and liquid phases. These results will be presented in upcoming publications.

ACKNOWLEDGMENTS

We thank Wouter-Jan Rappel for providing the Green's function steady-state code used to test some of our simulations, and Alain Karma for generously providing us with his unpublished results. We also thank Robert Almgren and Alain Karma for helpful discussions of our results at low undercooling. This work has been supported by the NASA Microgravity Research Program, under Grant NAG8-1249. We also acknowledge the support of the National Center for Supercomputing Applications (NCSA) for the use of its computer resources in producing some of our data.

REFERENCES

-
- [1] S.-C. Huang and M.E. Glicksman, *Acta. Metall.*, **29**, 701 (1981)]
 - [2] M. E. Glicksman, *Materials Science and Engineering*, **65** (1984).
 - [3] M. E. Glicksman, *Microgravity News*, NASA **4**, 4, (1997).
 - [4] J.S. Langer, *Rev. Mod. Phys.* **52**, 1 (1980).
 - [5] J.S. Langer, "Lectures in the Theory of Pattern Formation", in *Chance and Matter*, Les Houches Session XLVI, edited by J. Souletie, J. Vannimenus and R. Stora (North Holland, Amsterdam, 1987), p. 629.
 - [6] D.A. Kessler, J. Koplik and H. Levine, *Adv. Phys.* **37**, 255 (1988).
 - [7] G. P. Ivantsov, *Dokl. Akad. Nauk USSR*, **58**, 1113 (1947).
 - [8] G. Horvay and J. W. Cahn, *Acta Metall.* **9**, 695 (1961).
 - [9] D. E. Temkin, *Dokl. Akad. Nauk SSSR* **132**, 1307 (1960).
 - [10] R. Brower, D. Kessler, J. Koplik, and H. Levine, *Phys. Rev. Lett.*, **51**, 1111, (1983).
 - [11] E. Ben-Jacob, N. Goldenfeld, J.S. Langer and G. Schön, *Phys. Rev. Lett.*, **51**, 1930 (1983).
 - [12] E. Ben-Jacob, N. Goldenfeld, B.G. Kotliar, and J.S. Langer, *Phys. Rev. Lett.*, **53**, 2110 (1984).
 - [13] D. Kessler, J. Koplik, and H. Levine, *Phys. Rev. A*, **30**, 3161 (1984).
 - [14] E. Brener, and V. I. Melnikov, *Adv. Phys.* **40**, 53 (1991).
 - [15] Y. Pomeau and M. Ben Amar, *Dendritic growth and related topics*, in *Solids far from equilibrium*, ed. C. Godrèche, (Cambridge, 1991) 365.
 - [16] J. S. Langer, *Phys. Rev. A*, **36**, 3350 (1987).
 - [17] A. Karma, *Phys. Rev. E* **48**, 3441 (1993).
 - [18] E. Brener, and D. Temkin, *Phys. Rev. E*, **51**, 351 (1995).
 - [19] R. Almgren, *J. Comp. Phys.* **106**, 337 (1993).
 - [20] D. Juric, and G. Tryggvason, *J. Comp. Phys.* **123**, 127 (1996).
 - [21] J. S. Langer, in *Directions in Condensed Matter* (World Scientific, Singapore, 1986), 164.
 - [22] G. Caginalp, *Arch. Rat. Mech. Anal.* **92**, 205 (1986); G. Caginalp, *Anal. of Phys.* **172**, 136 (1986); G. Caginalp, and E. Socolovsky, *SIAM J. Sci. Comp.* **15**, 106 (1991).
 - [23] J. B. Collins and H. Levine, *Phys. Rev. B*, **31**, 6119 (1985).
 - [24] J. A. Warren and W. J. Boettinger, *Acta Metall. Mater.* **43**, 689 (1995)
 - [25] A. A. Wheeler, W.J. Boettinger, and G. B. McFadden *Phys. Rev. A* **45**, 7424 (1992) (1996).
 - [26] A. Karma, *Phys. Rev. E* **49**, 2245 (1994).
 - [27] K. R. Elder, F. Drolet, J. M. Kosterlitz, and M. Grant, *Phys. Rev. Lett.* **72**, 677 (1994).
 - [28] A. A. Wheeler, G. B. McFadden, and W.J. Boettinger, *Proc. Royal Soc. London A* **452**,
 - [29] R. Kobayashi, *Physica D* **63**, 410 (1993).
 - [30] N. Provatas, E. Elder, M. Grant, *Phys. Rev. B*, **53**, 6263 (1996).
 - [31] S.-L. Wang and R. F. Sekerka, *Phys. Rev. E* **53**, 3760 (1996).
 - [32] A. Karma, and W.-J. Rappel, *Phys. Rev. E* **53**, 3017 (1995); A. Karma, and Wouter-Jan Rappel, Preprint (1997).
 - [33] M. Fabbri and V. R. Voller, *J. Comp. Phys.* **130**, 256 (1997).
 - [34] G. Caginalp and X. Chen, *On The Evolution Of Phase Boundaries*, edited by M.E. Gurtin and G.B. McFadden (Springer-Verlag, New York, 1992), p.1.
 - [35] R. Almgren, Preprint (1997), available at "<http://www.cs.uchicago.edu/~almgren/pubs.html>".
 - [36] S. Chen, B. Merriman, S. Osher, and P. Smereka, *J. Comp. Phys.*, **135**, 8 (1997).
 - [37] B. Merriman, R. Caflisch, and S. Osher preprint (1998).
 - [38] R. J. Braun, and B. T. Murray, *J. Cryst. Growth* **174**, 41 (1997).
 - [39] H. J. Neeman, Ph.D Thesis, University of Illinois at Urbana-Champaign, Urbana, IL (1996).
 - [40] A. Schmidt, *J. Comp. Phys.* **125**, 293 (1996).
 - [41] N. Provatas, N. Goldenfeld, J. Dantzig, *Phys. Rev. Lett.*, **80**, 3308 (1998).
 - [42] P. R. B. Devloo, Ph.D Thesis, The University of Texas at Austin, Austin, TX (1987).
 - [43] M. S. Shephard, P. L. Baehmann, and K. R. Grice, *Comm. App. Num. Meth*, **4**, 379 (1988).
 - [44] N. Palle, and J. A. Dantzig, *Met. Trans. A*, **27A**, 707 (1996).
 - [45] R. Cook, D. Malkus, and M. Plesha, *Concepts and applications of finite element Analysis*, John Wiley and Sons, New York, NY, 1989.
 - [46] O. C. Zienkiewicz, and J. Z. Zhu, *Int. J. Numer. Meth.*

Eng., **24**, 337 (1987).

[47] R. Almgren, W. S. Dai and V. Hakim, Phys. Rev. Lett., **71**, 3461 (1993).

FIGURES

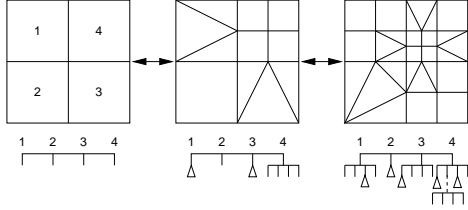


FIG. 1. An illustration of the quadtree element data structure. The first frame shows an element, and four child elements. Splitting of one of the children and one its children is shown, along with the branch evolution of the quadtree. Branches with triangles indicate square elements which are bridged with triangular or rectangular elements.

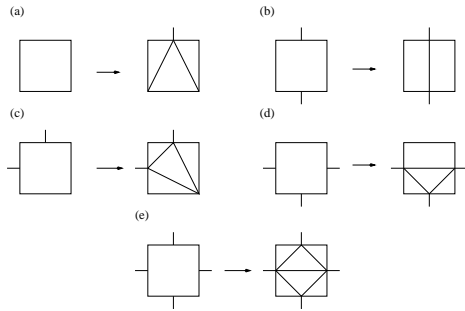


FIG. 2. Illustration of all possible configurations requiring completion with triangular and/or rectangular elements.

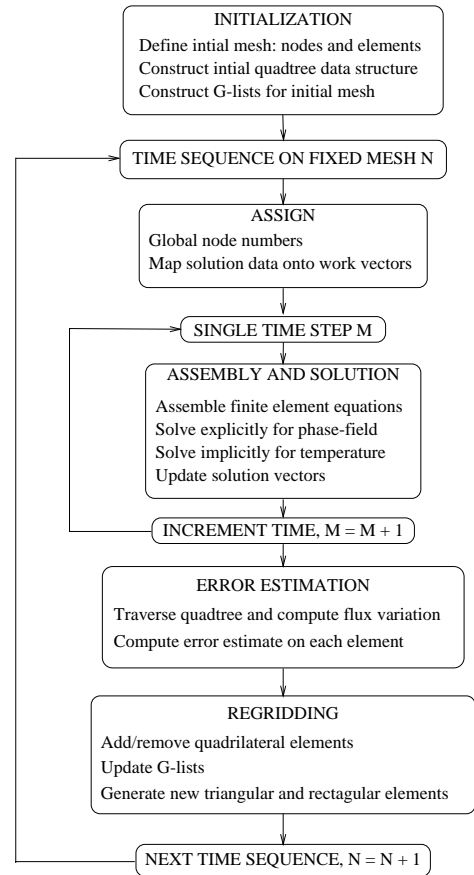


FIG. 3. A flow chart illustrating the algorithm program modules.

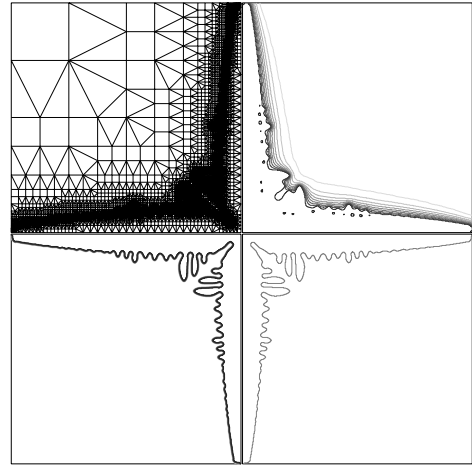


FIG. 4. A dendrite grown using the adaptive-grid method for $\Delta = 0.7$, $D = 2$, $\epsilon = 0.05$. Clockwise, beginning at the upper right the figures show contours of the U -field, the contour $\phi = 0$, contours of the ϕ -field and the current mesh.

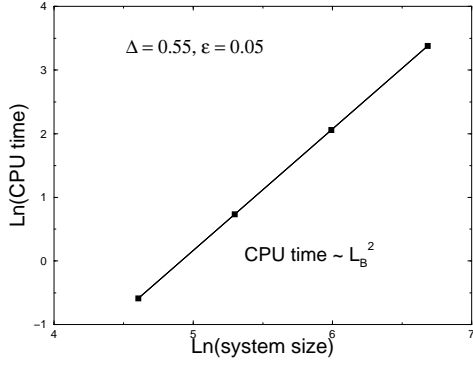
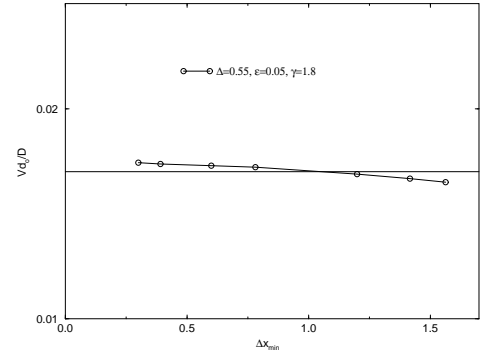


FIG. 5. CPU time vs. the system size, illustrating the computing time for a dendrite to move through the system of linear dimension L_B using our adaptive mesh method.



mi
dt

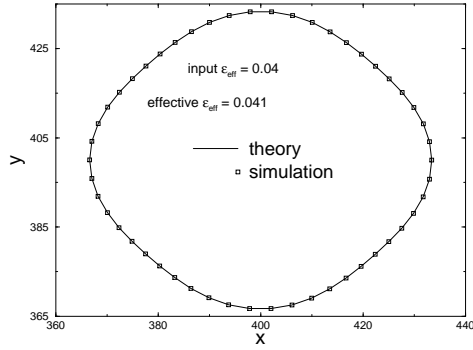


FIG. 6. The equilibrium shape of the interface, for an input anisotropy $\epsilon = 0.04$. The measured effective anisotropy $\epsilon = 0.041$.

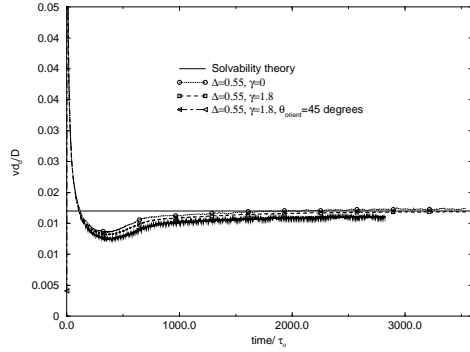


FIG. 7. The time evolution of the tip-velocity of a dendrite growing in the presence of surface tension anisotropy for $\Delta = 0.55$. Data is shown for the cases where the dendrite is moving in the x -direction with two grid layering patterns, and along the 45 degree line. The horizontal solid line represents the analytic prediction of microscopic solvability.

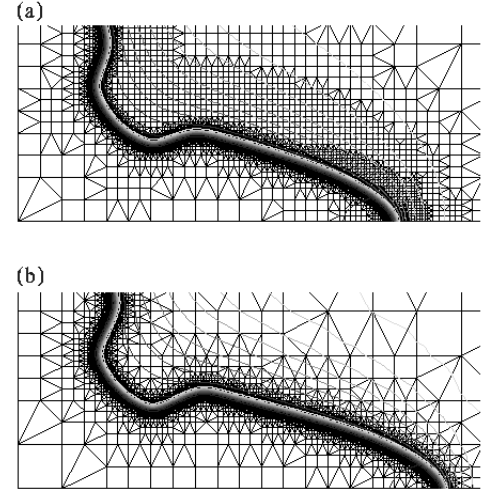


FIG. 9. The finite element mesh around a dendrite branch growing at $\Delta = 0.65$, showing the grid configuration for (a) $\gamma = 4$ and (b) $\gamma = 0$. The grey-shaded lines represent isotherms ranging from $-0.65 \leq U \leq 0.02$.

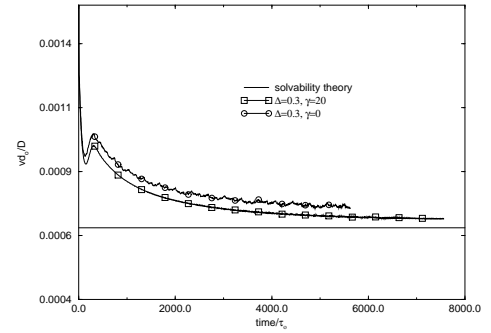


FIG. 10. The tip-velocity of a dendrite for $\Delta = 0.3$. Data are shown for two grid layering patterns. The horizontal solid lines represent the analytic prediction of microscopic solvability.

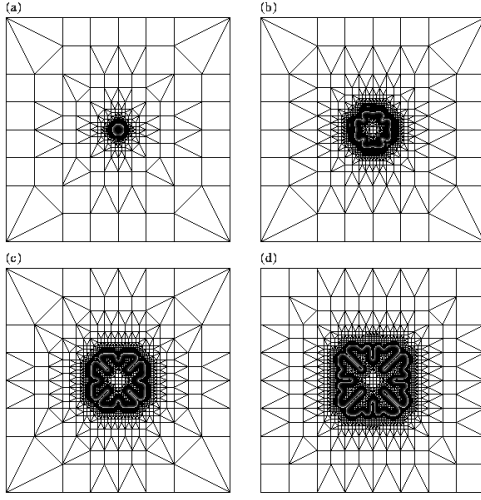


FIG. 11. The evolution of a crystal growing without interfacial anisotropy. The $\phi = 0$ contours are shown, superimposed on the finite element grids. Time advances from left to right, top to bottom.

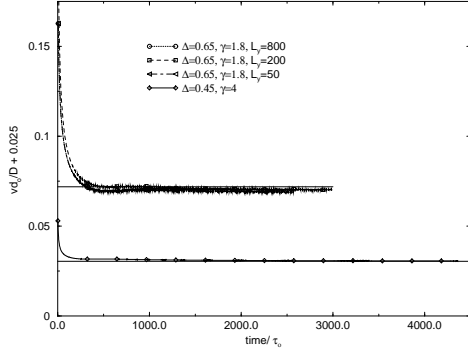


FIG. 12. The time evolution of the dimensionless tip velocity for $\Delta = 0.45$ and 0.65 . The horizontal lines represent solvability theory. The $\Delta = 0.65$ case includes data for three system sizes. The data have been shifted up by 0.025 for clarity.

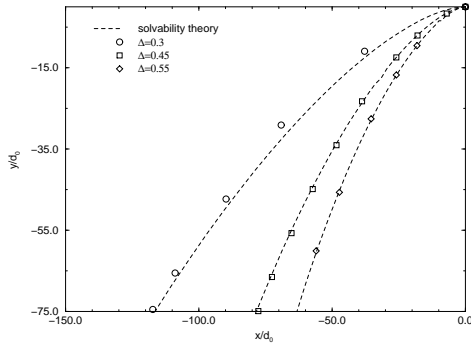


FIG. 13. The asymptotic dendrite tip shapes for $\Delta = 0.3$, 0.45 and 0.55 (data points). The dashed lines are the shapes predicted by solvability theory.

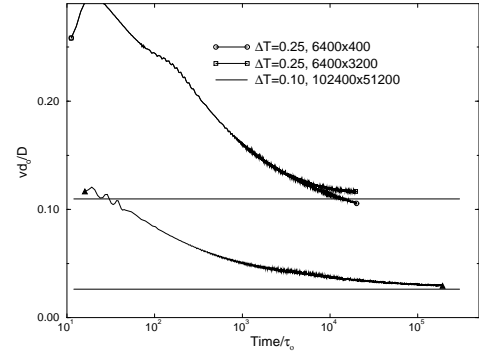


FIG. 14. The time evolution of the tip-velocity for undercooling $\Delta = 0.25$ and 0.10 . The data have been shifted up by 0.025 for clarity.

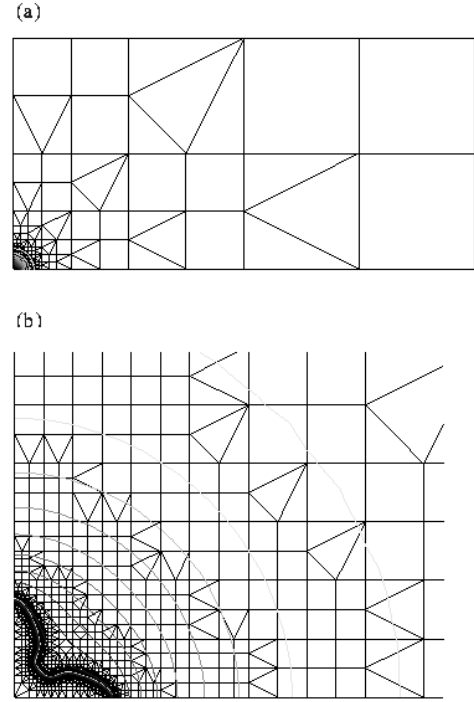


FIG. 15. Dendrite mesh and isotherms for undercooling $\Delta T = 0.1$. (a) shows the full domain whose dimensions are $102,400 \times 51,200$. The growing dendrite is in the lower left corner. (b) a close up displaying the dendrite tips, approximately $1,300$ units from the origin, while the temperature field spreads to about $5,000$ units.